Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»
Кафедра «Системы обработки информации и управления»
Курс «Базовые компоненты интернет-технологий»

Отчёт по Рубежному Контролю №2
по курсу «Базовые компоненты и интернет-технологии»
Вариант 16

Выполнил:
студент группы ИУ5-33Б
Рыбин Владислав

Проверил:
к.т.н., доц., Ю. Е. Гапанюк

2022 г.

Полученное задание:

1) Провести рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

2) Для текста программы рубежного контроля №1 создать модульные тесты с применением TDD - фреймворка (3 теста).

Текст программы:

```
from unittest import TestCase, main
from rk1 import *

class order_test(TestCase):
    def test_1(self):
        expected_result_for_C1 = [('Антикварная лавка', ['Курс Python', 'Анна
Каренина']),
                                  ('Аллегория', ['Евгений Онегин'])]
        res_C1 = []
        for i in Shops:
            book = list(filter(lambda a: a[2] == i.name, one_to_many))
            if i.name[0] == 'A':
                s_name = [x for x, _, _ in book]
                res_C1.append((i.name, s_name))
        self.assertEqual(res_C1, expected_result_for_C1)

    def test_2(self):
        expected_result_for_C2 = [('Книжный лабиринт', 100),
                                  ('Антикварная лавка', 179),
                                  ('Читай-город', 254),
                                  ('Аллегория', 660),
                                  ('Достоевский', 700)]
        res_unsorted_C2 = []
        for s in Shops:
            s_book = list(filter(lambda i: i[2] == s.name, one_to_many))
            if len(s_book) > 0:
                prices = [price for _, price, _ in s_book]
                prices_min = min(prices)
                res_unsorted_C2.append((s.name, prices_min))
        res_C2 = sorted(res_unsorted_C2, key=itemgetter(1))
        self.assertEqual(res_C2, expected_result_for_C2)

    def test_3(self):
        expected_result_for_C3 = [('Книжный лабиринт', 'Анжелика'),
                                  ('Антикварная лавка', 'Анна Каренина'),
                                  ('Достоевский', 'Всадник без головы'),
                                  ('Аллегория', 'Евгений Онегин'),
                                  ('Антикварная лавка', 'Курс Python'),
                                  ('Читай-город', 'Левша'),
                                  ('Читай-город', 'Повесть временных лет')]

        # Перебираем все отделы
        many_to_many.sort(key=lambda i: i[1])
        self.assertEqual(many_to_many, expected_result_for_C3)


if __name__ == '__main__':
    main()
```

Результаты выполнения: